Distributed Control by Lagrangian Steepest Descent

David H. Wolpert and Stefan Bieniawski

Abstract— **Often adaptive, distributed control can be viewed as an iterated game between independent players. The coupling between the players' mixed strategies, arising as the system evolves from one instant to the next, is determined by the system designer. Information theory tells us that the most likely joint strategy of the players, given a value of the expectation of the overall control objective function, is the minimizer of a Lagrangian function of the joint strategy. So the goal of the system designer is to speed evolution of the joint strategy to that Lagrangian minimizing point, lower the expectated value of the control objective function, and repeat. Here we elaborate the theory of algorithms that do this using local descent procedures, and that thereby achieve efficient, adaptive, distributed control.**

I. INTRODUCTION

It turns out that one can translate many of the probability-based concepts from statistical physics, game theory, distributed optimization and distributed control into one another. This translation allows one to transfer theory and techniques between those fields, creating a large common mathematics that connects them. This common mathematics is known as Probability Collectives. It concerns the set of probability distributions that govern any distributed system, and how to manipulate those distributions to optimize one or more objective functions [1].

This paper motivates Probability Collectives as a first-principles approach to adaptive distributed control problems [2], [3], [4]. To do this we represent such problems by having each control agent i set its state $x_i^t \in X$ independently of the other agents at each time t, by sampling an associated distribution, $q_i^t(x_i^t)$. In this representation the coupling between the agents does not arise directly, via statistical dependencies of the agents' states at the same time t. Rather it arises indirectly, through the stochastic joint evolution of their distributions $\{q_i^t\}$ across time.

More formally, let time be discrete, where at the beginning of each t all control agents simultaneously and independently set their states ("make their moves") by sampling their associated distributions. After they do so any remaining portions of the system (i.e., any stochastic part not being directly set by the control agents) responds to that joint move. Indicate the state of the entire system at time t as z^t. (z^t includes the joint move of the agents, x^t, as well as the state at t of all stochastic elements not directly set by the agents.) So the joint distribution of the moves of the agents at any moment t is given by the product distribution $q^t(x^t) = \prod_i q_i^t(x_i^t)$, and the state of the entire system, given joint move x^t, is governed by $P(z^t \mid x^t)$.

Now in general the observations by agent i of aspects of the system's state at times previous to t will determine q_i^t. In turn, those observations are determined by the previous states of the system. So q_i^t is statistically dependent on the previous states of the entire system, $z^{\{t' < t\}}$. Accordingly, the system can be viewed as a multi-stage noncooperative game among the agents and Nature. Each agent plays mixed strategies $\{q_i^t\}$ at moment t, and Nature's move space at that time consists of those components of the vector z^t not contained in x^t [5], [6], [7], [8], [9]. The interdependence of the agents across time can be viewed as arising through information sets and the like, as usual in game theory.

For pedagogical simplicity, consider the problem of inducing an optimal state z rather than the problem of inducing an optimal sequence of states.[1] What the designer of the system can specify are the laws that govern how the joint mixed strategy q^t gets updated from one stage of the game to the next. The goal is to specify such laws that will quickly lead to a good value of an overall objective function of the state of the system, $F(z)$.[2] Note that the agents work in the space X; all aspects of the system not directly set by the agents, and in particular all noise processes, are implicitly contained in the distribution $P(z \mid x)$. Tautologically then, in distributed control the goal is to induce a joint strategy $q(x)$ with a good associated value of $E_q(F) = \int dx q(x) E(F \mid x) \triangleq \int dx q(x) G(x) = E_q(G)$.[3] Once such a q is found, one can sample it to get a final x, and be assured that, on average, the associated F value is low. G is called the **world utility**.

In this paper we elaborate an iterative scheme for updating product distributions q to monotonically lower $E_q(G)$. Each q in the sequence is defined indirectly, as the minimizer of a different G-parameterized Lagrangian, $L_G(q)$. Implementing such a sequence of Lagrangian-minimizing q's results in the optimal control policy for the distributed system, i.e., in the q minimizing $E_q(G)$. However while one cannot directly solve for the q minimizing $E_q(G)$ in a distributed manner, as elaborated below one can solve for the q minimizing each $L_G(q)$ in a distributed manner. In this way one can find the optimal distributed control policy using a purely distributed algorithm.

The distributed algorithms presented here are all based

D. Wolpert is with NASA Ames Research Center, Moffett Field, CA, 94035 dhw@ptolemy.arc.nasa.gov

S. Bieniawski is with Stanford University, Stanford, CA, 94305 stefanb@stanford.edu

[1] See [10] for a discussion of the problem of optimizing a sequence.

[2] Here we follow the convention that lower F is better. In addition, for simplicity we only consider objectives that depend on the state of the system at a single instant; it is straightforward to relax this restriction.

[3] For simplicity, here we indicate integrals of any sort, including point sums for countable X, with the $\int$ symbol.

on using steepest descent-type techniques to minimize each successve Lagrangian.[4] Because the descent is over Euclidean vectors q, these algorithms can be applied whether the x_i are categorical/symbolic, continuous, time-extended, or a mixture of the three. So in particular, they provide a principled way to do "gradient descent over categorical variables".

In the next section we derive the functional form of the Lagrangians $L_G(q)$ and discuss some of its properties.

In the following section we show how to apply gradient descent (and its embellishments) to optimize the Lagrangian in a distributed fashion. If we view the agents as engaged in a team game, all having the same utility G, then this gradient descent is a distributed scheme for each agent to update its strategy, in a way that will steer the game to a bounded rational equilibrium [1], [11].

In this section we also consider second order methods. In contrast to gradient descent, in general any single application of Newton's method to update a product distribution q will result in a new distribution p_q that is not a product distribution. So we must instead solve for the product distribution $q'(p_q)$ having minimal Kullback-Leibler distance to p_q. In this section we derive the rule for iterative updating of our distribution so as to move q in the direction of $q'(p_q)$. Serendipitously, this rule can also be implemented in a distributed fashion.

In practice any local descent scheme often requires Monte Carlo sampling to estimate terms in the gradient. To minimize the expected quadratic error of the estimation, typically the game is changed from being a team game. In other words, in general changing the agent's utilities g_i to not all equal G will result in lower bias plus variance of the estimation of the gradient, and therefore will speed evolution to a good joint strategy. There are many other techniques as well for improving the Monte Carlo sampling. These include data-aging, and techniques for managing the descent when it gets close to a border of the space of product distributions. Most of these techniques can be used even with schemes for minimizing $L_G(q)$ other than gradient descent. These are cursorily discussed at the end of this section. See[13], [16] for more details.

In the final section we introduce some alternatives to $L_G(q)$, designed to help speed convergence to a q with low $E(G)$. Miscellaneous proofs can be found in the appendix.

The version of Probability Collectives considered in this paper, involving product distributions, is called "Product Distribution" (PD) theory[13]. Some initial experimental results concerning the use of PD theory for distributed optimization and distributed control can be found in [17], [18], [19], [20], [21], [22]. See [15], [12], [10] for other uses and extensions of PD theory.

[4]See [11], [12], [13], [14] for non-local techniques for finding q^t, techniques that are related to ficticious play, and see [15] for techniques that exploit the Metropolis-Hastings algorithm. Other non-local techniques are related to importance sampling of integrals, and are briefly mentioned in [1].

II. Product Distribution Lagrangians

A. The Maxent Lagrangian

Say the designer stipulates a particular desired value of $E(G)$, γ. For simplicity, consider the case where the designer makes no other claims concerning the system besides γ and the fact that the joint strategy is a product distribution. Then information theory tells us that the *a priori* most likely q consistent with that information is the one that maximizes entropy subject to that information [23], [24], [25].[5] In other words, of all distributions that agree with the designer's information, that distribution is the "easiest" one to induce by random search.

Given this, one can view the job of the designer of a distributed control system as an iterative equilibration process. In the first stage of each iteration the designer works to speed evolution of the joint strategy to the q with maximal entropy subject to a particular value of γ. Once we have found such a solution we can replace the constraint — replace the target value of $E(G)$ — with a more difficult one, and then repeat the process, with another evolution of q [13].

To formalize this, define the **Maxent** Lagrangian by

$$\begin{aligned} L(q) \triangleq L_\gamma(q) &\triangleq \beta(E_q(G) - \gamma) - S(q) \\ &= \beta(\int dx q(x) G(x) - \gamma) - S(q) \end{aligned} \quad (1)$$

where $S(q)$ is the Shannon entropy of q, $-\int dx q(x) \ln \frac{q(x)}{\mu(x)}$, and for simplicity we here take the prior μ to be uniform.[6]. Given γ, the associated most likely joint strategy is the q that minimizes $L(q)$ over all those (q, β) such that the Lagrange parameter β is at a critical point of L_γ, i.e., such that $\frac{\partial L}{\partial \beta} = 0$.

Solving, we find that the q_i are related to each other via a set of coupled Boltzmann equations (one for each agent i),

$$q_i^\beta(x_i) \propto e^{-\beta E_{q_{(i)}^\beta}(G|x_i)} \quad (2)$$

where the overall proportionality constant for each i is set by normalization, the subscript $q_{(i)}^\beta$ on the expectation value indicates that it is evaluated according to the distribution $\prod_{j \neq i} q_j$, and β is set to enforce the condition $E_{q^\beta}(G) = \gamma$. Following Nash, we can use Brouwer's fixed point theorem to establish that for any fixed β, there must exist at least one solution to this set of simultaneous equations.

In light of the foregoing, one natural choice for an algorithm that lowers $E_q(G)$ is the repeated iteration of the following step: Start with the q^β matching a current γ value, then lower γ slightly, and end by modifying the old q^β to

[5]In light of how limited the information is here, the algorithms presented below are best-suited to "off the shelf" uses; incorporating more prior knowledge allows the algorithms to be tailored more tightly to a particular application.

[6]Throughout this paper the terms in any Lagrangian that restrict distributions to the unit simplices are implicit. The other constraint needed for a Euclidean vector to be a valid probability distribution is that none of its components are negative. This will not need to be explicitly enforced in the Lagrangian here.

find the one that matches the new γ. A difficulty with this iterative step is the need to solve for β as a function of γ. However we can use a trick to circumvent this need. Typically if we evaluate $E(G)$ at the solutions q^β, we find that it is a declining function of β. So in following the iterative procedure of equilibrating and then lowering γ we will raise β. Accordingly, we can avoid the repeated matching of β to each successive constraint $E(G) = \gamma$, and simply monotonically increase β instead. This allows us to avoid ever explicitly specifying the values of γ (see appendix).

An alternative interpretation of this iterative scheme is based on prior knowledge of the value of the entropy rather of the expected G. Given this alternative prior knowledge, we can recast the designer's goal as finding the q that is consistent with that knowledge that has minimal $E(G)$. This again leads to Eq.'s 1 and 2. Now raising β is cast as lowering the (never-specified) prior knowledge of the entropy value rather than the (never-specified) prior knowledge of $E(G)$.

Simulated annealing is an example of this approach, where rather than work directly with q, one works with random samples of it formed via the Metropolis random walk algorithm [26], [27], [28], [29]. There is no *a priori* reason to use such an inefficient means of manipulating q however. Here we will work with q directly instead. This will result in an algorithm that is not simply "probabilistic" in the sense that the updating of its variables is stochastic (as in simulated annealing). Rather the very entity being updated is a probability distribution.

Another advantage of casting the problem directly in terms of the Maxent Lagrangian is that one can even avoid the need to explicitly stipulate an annealing schedule. In the usual way, first order methods can be used to find the saddle point of the Lagrangian, e.g., by performing steepest ascent of L in the Lagrange parameter β while performing a descent in q [7].

In many situations one should use a modification of the Maxent Lagrangian. Whenever one has extra prior knowledge about the problem domain, that should be used to modify the use of entropy as (in statistics terminology) a regularizer. This leads to Bayesian formulations [30]. Similarly, if one has constraints $\{f_i(x) = 0\}$, the Lagrangian has to be modified to account for them. The most naive way of doing this is to simply cast the constraints as Lagrange penalty terms $\{E(f_i) = 0\}$ and add those terms to the Lagrangian, in the usual way [30], [22] [8].

[7]Formally, since the Maxent Lagrangian is not convex, we have no guarantee that the duality gap is zero, and therefore no guarantee about saddlepoints. Nonetheless, just as in other domains, first order methods here seem to work well in practice.

[8]Note though that since the gradient of entropy is infinite at the border of the unit simplex, we are guaranteed that no component of q will ever exactly equal 0, which typically means that the constraints $\{f_i(x) = 0\}$ will never be satisfied with probability exactly 1.

B. Geometry of the Maxent Lagrangian

To investigate the geometry of the Maxent Lagrangian we must be careful to distinguish between the various possible parameterizations of distributions. To begin, let $\mathcal{P}$ be the space of distributions (product or otherwise) over our variables. In addition define $\mathcal{Q} \subset \mathcal{P}$ as the set of product distributions over X. Also for all $i = 1, \ldots..., n$, define $o(X_i)$ as the number of possible values x_i (for simplicity taken to be finite), and write $o(X) = \prod_i o(X_i)$. So the simplex $\mathcal{P} \subset \mathbb{R}_+^{o(X)}$.

Let t be an arbitrary element of $\mathbb{R}_+^{o(X)}$, not necessarily normalized. The entropy with its domain extended to all t, $-\int dx\, t(x) \ln t(x)$, is a concave function over $\mathbb{R}_+^{o(X)}$. Therefore it is also concave over any subset of $\mathbb{R}_+^{o(X)}$, and in particular the entropy of normalized vectors $p \in \mathcal{P}$ with components $p(x)$ is concave. In addition $\int dx\ G(x)t(x)$ is a convex function of t. So the Maxent Lagrangian $L(p \in \mathcal{P})$ is a convex function. Since $\mathcal{P}$ is a convex space, this means the Maxent Lagrangian has at most one minimum over $\mathcal{P}$.

However we are restricting attention to $\mathcal{Q}$ (to reduce the number of parameters we're going to be using, if for no other reason). In general this is not a convex subspace of $\mathcal{P}$; if $p \in \mathcal{Q}$ and $p' \in \mathcal{Q}$, then distributions on the line connecting p and p' will still lie in $\mathcal{P}$ but may not lie in $\mathcal{Q}$. So even though the Maxent Lagrangian is convex over $\mathcal{Q}$, we do not have guarantees of a single local minimum.

There are other ways to parameterize product distributions however rather than as points in $\mathcal{P} \subset \mathbb{R}_+^{o(X)}$. In particular, consider the space $\mathbb{R}_+^{\sum_i o(X_i)}$. The most appropriate way to express a product distribution q is as a vector in this space, since that way we can assign each separate distribution q_i to a different set of components. For example, we can parameterize any $q \in \mathcal{Q}$ using $\mathbb{R}_+^{\sum_i o(X_i)}$ via the **Naive Distributed Parameterization** (NDP), $q(x) = \prod_i q_i(x_i)$. Note that this parameterization of $\mathcal{Q}$ is many-to-one.[9]

Unlike when it is parameterized as a subset of $\mathbb{R}_+^{o(X)}$, $\mathcal{Q}$ is convex when parameterized as a subset of $\mathbb{R}_+^{o(X_i)}$ under the NDP; it is the set of $\{q_i\}$ such that $\int dx_i\ q_i(x_i) = 1\ \forall i$, i.e., the n-fold Cartesian product of unit simplices. In addition, say we express the entropy via the NDP as $S(\{q_i\}) = -\int dx\, [\prod_i q_i(x_i)] \ln(q_i(x_i))$. If we restrict each q_i to be normalized, then $S(\{q_i\}) = -\sum_i \int dx_i\ q_i(x_i) \ln(q_i(x_i))$, and is concave, as before.

As an aside, note that if we allow $\{q_i\}$ to range over all $\mathbb{R}_+^{\sum_i o(X_i)}$, then the extension of entropy as just defined is not concave in general (unlike in the case for the extension of the definition of entropy of points $p \in calP$. In general,

$$-\int dx\, [\prod_i q_i(x_i)] \ln(q_i(x_i)) \quad \neq \quad -\sum_i \int dx_i\ q_i(x_i) \ln(q_i(x_i))$$

[9]In particular, a vector $\vec{q} \in \mathbb{R}_+^{\sum_i o(X_i)}$ with components $\vec{q}_x$ may obey $\sum_x \vec{q}_x = 1$ even though for some i-specified subset of components, $\{\vec{q}_{x_i}\}$, $\sum_{x_i} \vec{q}_{x_i} \neq 1$. As an example, say we have two agents each of whom can make two moves. Then both $[q_1 = (2/3, 1/3), q_2 = (1/2, 1/2)]$ and $[q_1 = (2, 1), q_2 = (1/6, 1/6)]$ have the same image under the NDP, $q = (1/3, 1/3, 1/6, 1/6)$.

for non-normalized $\{q_i\}$, the Hessian of S has off-diagonal entries, and its eigenvalues can have mixed signs. To avoid ever confronting this issue, it is expedient to simply choose a different way of extending the domain of the entropy function under the NDP, namely

$$S(\{q_i\}) \triangleq -\sum_i \int dx_i \; q_i(x_i) \ln(q_i(x_i)).$$

Return now to the issue of the convexity of our particular (normalized distributions) optimization problem under the NDP. While with the NDP our feasible set is convex as desired, and the entropy term of the Maxent Lagrangian is concave as desired, now expected G presents problems: $\int dx \, G(x) \prod_i q_i(x_i)$ is a multilinear function of the components of $\{q_i\}$, and therefore is not convex.

We can fix this while still using the space $\mathbb{R}_+^{\sum_i o(X_i)}$, so long as we change from the NDP. For example, since geometric means are concave functions, $\int dx \prod_i [r_i(x_i)]^{1/n} G(x)$ is a convex function of $\{r_i\}$, so long as $G(x) \leq 0 \; \forall x$. Unfortunately, in this new parameterization the subtracted entropy term in the Lagrangian, $\sum_i \int dx_i \; [r_i(x_i)]^{1/n} \ln([r_i(x_i)]^{1/n})$, is concave rather than convex. A more important problem is that the set of r such that $\prod_i [r_i(x_i)]^{1/n}$ is normalized is not convex. So for simplicity, from now on we will use the NDP, and write "q" as shorthand for $\{q_i\}$.

Consider L as a function of q, with β and γ both treated as fixed parameters. (So in particular, $E_q(g)$ need not equal γ.) First, say that $q_{(i)}$ is also held fixed, with only q_i allowed to vary. This makes $E(g)$ linear in q_i. In addition, entropy is a concave function, and the unit simplex is a convex region. Accordingly, the Lagrangian of Eq. 1 has a unique local minimum over q_i; there is no problem of "getting trapped in a local minimum" in a computational search for that minimum. Indeed, in this situation we can just jump directly to that global optimum, via Eq. 2.

Now introduce the shorthand for any function $U(x)$,

$$[U]_{i,p}(x_i) \triangleq \int dx_{(i)} U(x_i, x_{(i)}) p(x_{(i)} \mid x_i).$$

So $[G]_{i,q_{(i)}}(x_i)$ is agent i's "effective" cost function, $E_{q_{(i)}}(G \mid x_i)$. Consider the value $E_{q_i^\beta}([G]_{i,q_{(i)}})$. This is the value of $E(G)$ at i's bounded rational equilibrium for the fixed $q_{(i)}$, i.e., it is the value at the minimum over q_i of L. View that value as a function of β. One can show that this is a decreasing function. In fact, its derivative just equals the negative of the variance of $[G]_{i,q_{(i)}}(x_i)$ evaluated under distribution $q_i^\beta(x_i)$ (see appendix). Combining this with the fact that $E(G)$ is bounded below (for bounded G), establishes that the variance must go to zero for large enough β. So as β grows, $q_i^\beta(x_i) \to 0$ for all x_i that don't minimize $E_{q_{(i)}}(G \mid x_i)$. In other words, in that limit, q_i becomes Nash-optimal.

The following lemma extends the technique of Lagrange parameters to off-equilibrium points:

Lemma 1: Let $y' \in \mathbb{R}^n$ be a point consistent with a set of constraints over $\mathbb{R}^n$, $\{f_i(y) = 0\}$. Consider the set of all vectors leading from y' that are, to first order, consistent with those constraints. Of those vectors, the one giving the steepest ascent of a function $V(y)$ is $\vec{u} = \nabla V + \sum_i \lambda_i(y') \nabla f_i$, up to an overall proportionality constant, where the $\lambda_i(y')$ enforce the first order consistency conditions, $\vec{u} \cdot \nabla f_i = 0 \;\; \forall i$.

Now examine the derivatives of $S(q)$ with respect to all components of q, i.e., the q-gradient of the entropy. At the border of $\mathcal{Q}$, at least one of the $\ln(q_i)$ terms in those derivatives will be negative infinite. As explained in the appendix, combined with Lemma 1, this fact establishes that at the edge of $\mathcal{Q}$, the steepest descent direction of any player's Lagrangian points into the interior of $\mathcal{Q}$ (assuming finite β and $\{G\}$). (This is reflected in the equilibrium solutions Eq. 2.) Accordingly, whereas Nash equilibria can be on the edge of $\mathcal{Q}$ (e.g., for a pure strategy Nash equilibrium), in bounded rational games any equilibrium must lie in the interior of $\mathcal{Q}$. In other words, any equilibrium (i.e., any local minimum) of a bounded rational game has non-zero probability for all joint moves. So just as when only varying a single q_i, we never have to consider extremal mixed strategies in searching for equilibria over all $\mathcal{Q}$. We can use local descent schemes instead [17], [21], [30].

Lemma 1 can also be used to construct G with more than one solution to Eq. 2. One can also show that for every player i and any point q interior to $\mathcal{Q}$, there are directions in $\mathcal{Q}$ along which i's Lagrangian is locally convex. Accordingly, no player's Lagrangian has a local maximum interior to $\mathcal{Q}$. So if there are multiple local minima of i's Lagrangian, they are separated by saddle points across ridges. In addition, the uniform q is a solution to the set of coupled equations Eq. 2, but typically is not a local minimum, and therefore must be a saddle point.

Say that we were not restricting ourselves to product distributions. So the Lagrangian becomes $L(p) = \beta(E_p(G) - \gamma) - S(p)$, where p can now be any distribution over x. There is only one local minimum over p of this Lagrangian, the **canonical ensemble**:

$$p^\beta(x) \propto e^{-\beta G(x)}$$

In general p^β is not a product distribution. However we can ask what product distribution is closest to it.

Now in general, the proper way to approximate a target distribution p with a distribution from a subset C of the set of all distributions is to first specify a misfit measure saying how well each member of C approximates p, and then solve for the member with the smallest misfit. This is just as true when C is the set of all product distributions as when it is any other set.

How best to measure distances between probability distributions is a topic of ongoing controversy and research [31]. The most common way to do so is with the infinite limit log likelihood of data being generated by one distribution

but misattributed to have come from the other. This is know as the **Kullback-Leibler distance** [23], [32], [24]:

$$KL(p^1 \parallel p^2) \triangleq S(p^1 \parallel p^2) - S(p^1) \tag{3}$$

where $S(p^1 \parallel p^2) \triangleq -\int dx\ p^1(x) \ln[\frac{p^2(x)}{\mu(x)}]$ is known as the **cross entropy** from p^1 to p^2 (and as usual we implicitly choose uniform μ).

The KL distance is always non-negative, and equals zero iff its two arguments are identical. In addition, $KL(\alpha p^1 + (1-\alpha)p^2 \parallel p^2)$ is an increasing function of $\alpha \in [0.0, 1.0]$, i.e., as one moves along the line from p^1 to p^2, the KL distance from p^1 to p^2 shrinks.[10] The same is true for $KL(p^2 \parallel \alpha p^1 + (1-\alpha)p^2)$. In addition, those two KL distances are identical to 2nd order about $\alpha = 0$. However they differ as one moves away from $\alpha = 0$ in general; KL distance is not a symmetric function of its arguments. In addition, it does not obey the triangle inequality, although it obeys a variant [23]. Despite these shortcomings, it is by far the most common way to measure the distance between two distributions.

As shorthand, define the "pq distance" as $KL(p \parallel q)$, and the "qp distance" as $KL(q \parallel p)$, where p is our target distribution and q is a product distribution. Then it is straightforward to show that the qp distance from q to target distribution p^β is just the Maxent Lagrangian $L(q)$, up to irrelevant overall constants. In other words, the q minimizing the Maxent Lagrangian is q with the minimal qp distance to the associated canonical ensemble.

However the qp distance is the (infinite limit of the negative log of) the likelihood that distribution p would attribute to data generated by distribution q. It can be argued that a better measure of how well q approximates p would be based on the likelihood that q attributes to data generated by p. This is the pq distance; it gives a different Lagrangian from that of Eq. 1.

Evaluating, up to an overall additive constant (of the canonical distribution's entropy), the pq distance is

$$KL(p \parallel q) = -\sum_i \int dx\ p(x) \ln[q_i(x_i)].$$

This is equivalent to a game where each coordinate i has the "Lagrangian"

$$L_i^*(q) \quad \triangleq \quad -\int dx_i\ p_i(x_i) \ln[q_i(_i)], \tag{4}$$

where $p_i(x_i)$ is the marginal distribution $\int dx_{(i)} p(x)$. The minimizer of this is just $q_i = p_i\ \forall i$, i.e., each q_i is set to the associated marginal distribution of p.

In most of this paper we restrict attention to the qp KL distance and associated Maxent Lagrangian.

[10] This follows from the fact that the second derivative with respect to α is non-negative for all α, combined with the fact that KL distance is never negative and equals 0 when $\alpha = 0$.

III. Descent of the Maxent Lagrangian

A. Gradient descent

Consider the situation where each x_i can take on a finite number of possible values, $|X_i|$. Say we are iteratively evolving q to minimize L for some fixed β, and are currently at some point $q \in \mathcal{Q}$. Using Lemma 1, we can evaluate the direction from q within $\mathcal{Q}$ that, to first order, will result in the largest drop in the value of $L(q)$:

$$\frac{\partial^R L(q)}{\partial^R q_i(x_i = j)} \quad = \quad u_i(j) - \sum_{x_i'} u_i(x_i')/|X_i|, \tag{5}$$

where $u_i(j) \triangleq \beta E(G \mid x_i = j) + \ln[q_i(j)]$, and the symbol ∂^R indicates that we do not mean the indicated partial derivative, formally speaking, but rather the indicated component of the 1st-order descent vector [11] Intuitively, the reason for subtracting $\sum_{x_i'} u_i(x_i')/|X_i|$ is to keep the distribution in the set of all possible probability distributions over x, $\mathcal{P}$. By Lemma 1, the update step must use such subtraction of a normalization vector. In particular, it is not correct to get back to the space of probability distributions by multiplying by an overall normalization constant.

Eq. 5 specifies the change that each agent should make to its distribution to have them jointly implement a step in steepest descent of the Maxent Lagrangian. These updates are completely distributed, in the sense that each agent's update at time t is independent of any other agents' update at that time. Typically at any t each agent i knows $q_i(t)$ exactly, and therefore knows $\ln[q_i(j)]$. However often it will not know G and/or the $q_{(i)}$. In such cases it will not be able to evaluate the $E(G \mid x_i = j)$ terms in Eq. 5 in closed form.

One way to circumvent this problem is to have those expectation values be simultaneously estimated by all agents by repeated Monte Carlo sampling of q to produce a set of $(x, G(x))$ pairs. Those pairs can then be used by each agent i to estimate the values $E(G \mid x_i = j)$, and therefore how it should update its distribution. In the simplest version of such an update to q only occurs once every L time-steps. In this scheme only the samples $(x, G(x))$ formed within a block of L successive time-steps are used at the end of that block by the agents to update their distributions (according to Eq. 5).

B. Higher order descent schemes

In general, second order descent (e.g., Newton's method) of the Maxent Lagrangian is non-trivial, due to coupling that arises between the agents and the requirement for associated matrix inversion. An alternative approach starts by making a quadratic approximation (over the space of all p, not just all q) to the Maxent Lagrangian, $L(p)$. Via Newton's method this specifies a p^* that minimizes that quadratic approximation. We can then find the product distribution that is nearest (in pq KL distance) to p^*. This scheme is called **Nearest Newton** descent.

[11] Formally speaking, the partial derivative is given by $u_i(j)$.

The gradient and Hessian of L at a current point p^0 are given by

$$\begin{aligned}\frac{\partial L}{\partial p(x)}|_{p=p^0} &= \beta G(x) + 1 + \ln(p^0(x)) \\ \frac{\partial^2 L}{\partial p(x)\partial p(x')}|_{p=p^0} &= \frac{\delta_{x,x'}}{p^0(x)}.\end{aligned}$$

This Hessian is positive-definite (given that $p^0 \in \mathcal{P}$). By simple Lagrange parameters, the (normalized) distribution that minimizes a paraboloid with this Hessian and gradient is either on the border of $\mathcal{P}$, or if in the interior is given by

$$p^*(x) = -p^0(x)\left[\beta G(x) + \ln(p^0(x)) + \lambda\right]$$

where λ is set by normalization. Solving, either p^* is on the edge of the simplex, or

$$\frac{p^*(x)}{p^0(x)} = 1 - S(p^0) - \ln(p^0(x)) - \beta[G(x) - E(G)]$$

where $E(G)$ is evaluated under p^0.

Note that the right-hand side is exactly the direction you should go using (simplex-constrained) gradient descent of $L(p)$. So the direction to p^* from p^0 is given by the Hadamard product of p^0 and the direction given by gradient descent.

Now we can approximate p^* with the product distribution having the minimal KL distance to it. In particular, consider using pq KL distance rather than qp KL distance. Recall that for this kind of KL distance, the optimal product distribution approximation to a joint distribution is given by the product of the marginals of that joint distribution (see the discussion just below Eq. 4). Say that p^0 is in the form of a product distribution, q^0, i.e., that we are starting from a product distribution. Then calculating the marginals of the associated p^* to get q^* is trivial:

$$\begin{aligned}\frac{q_i^*(j)}{q_i^0(j)} = 1 &- S(q_i^0) - \ln(q_i^0(j)) \\ &- \beta[E(G \mid x_i = j) - E(G)] \qquad (6)\end{aligned}$$

Note that since the original quadratic approximation was over the full joint space, this formula automatically takes into account inter-agent couplings. In practice of course, it may make sense not to jump all the way from q^0 to q^*, but only part-way there, to be conservative. (In fact, if q^* isn't in the interior of the simplex, such partial jumping is necessary.) One potential guide to how far to jump is the pq KL distance from p^* to $\prod_i q_i^*$. Unlike the KL distances to the full joint Boltzmann distribution, we can readily calculate this KL distance.

The conditional expectations in Nearest Newton are the same as those in gradient descent. Accordingly, they too can be estimated via Monte Carlo sampling, if need be. It's also worth noting that Eq. 6 has the same form as one would get by evaluating the Hessian of the Maxent Lagrangian, so long as one ignored inter-agent aspects of that Hessian.

C. Practical issues

In practice, the block-wise Monte Carlo sampling to estimate descent directions described above can be prohibitively slow. The estimates typically have high variance, and therefore require large block size N to get a good descent direction. One set of ways to address this is to replace the team game with a non-team game, i.e., for each agent i have it estimate quantities $E(g_i \mid x_i = j)$ rather than $E(G \mid x_i = j)$, where each **private utility** g_i is chosen to have much lower variance than G [13], [17], [12].[12]

Another useful technique is to allow samples from preceding blocks to be re-used. One does this by first "aging" that data to reflect the fact that it was formed under a different $q_{(i)}$. For example, one can replace the empirical average for the most recent block k,

$$\hat{G}_{i,j}(k) \triangleq \frac{\sum_{t=kN}^{kN+N} G(x^t)\delta_{x_i^t,j}}{\sum_{t=kN}^{kN+N} \delta_{x_i^t,j}}$$

with a weighted average over the expected G's of all preceding blocks,

$$\frac{\sum_m \hat{G}_{i,j}(m) e^{-\kappa(k-m)}}{\sum_m e^{-\kappa(k-m)}}$$

for some appropriate aging constant κ.[13]

Typically such aging allows N to be vastly reduced, and therefore the overall minimization of N to be greatly sped up. For such small N though, it may be that the most recent block has *no* samples of some move $x_i = j$. This would mean that $\hat{G}_{i,j}(k)$ is undefined. One crude way to avoid such problems is to simply force a set of samples of each such move if they don't occur of their own accord, being careful to have the $x_{(i)}$ formed by sampling $q_{(i)}$ when forming those forced samples. Another alternative is to average over just those k for which $G_{i,j}(k)$ exists.

Other useful techniques allow one to properly decrease the step size as one nears the border of $\mathcal{Q}$.

IV. Other Lagrangians For Finding Minima Of G

There are many alternative Lagrangians to the ones described above. The section focuses on such alternative Lagrangians for the purpose of finding $\text{argmin}_x G(x)$. Two classes of such Lagrangians are investigated: variants of the Maxent Lagrangians, and variants of the two types of KL-distance Lagrangians.

[12]Formally, this means that each agent i has a separate Lagrangian, formed from Eq. 1 by substituting g_i for G. The associated joint solution q is then given by substituting the appropriate g_i for G in each instance of the coupled equations Eq. 2 (one instance for each i). See [1] for the relation of this to bounded rational game theory.

[13]Not all preceding $\hat{G}_{i,j}(m)$ need to be stored to implement this; exponential ageing can be done online using 3 variables per (i, j) pair.

A. Maxent Lagrangians

Say that after finding the q that minimizes the Lagrangian, we IID sample that q, K times. We then take the sample that has the smallest G value as our guess for the x that minimizes $G(x)$. For this to give a low x we don't need the mean of the distribution $q(G)$ to be low — what we need is for the bottom tail of that distribution to be low. This suggests that in the $E(G)$ term of the Maxent Lagrangian we replace

$$q(x) \leftarrow q(x)\frac{\Theta[\kappa - \int dx'\ q(x')\Theta[G(x) - G(x')]]}{\kappa}$$

where Θ is the Heaviside theta function. This new multiplier of G is still a probability distribution over x. It equals 0 if $G(x)$ is in the worst $1-\kappa$ percentile (according to distribution q) of G values, and κ^{-1} otherwise. So under this replacement the $E(G)$ term in the Lagrangian equals the average of G restricted to that lower κ'th percentile. For $\kappa = K^{-1}$, our new Lagrangian forces attention in setting q on that outlier likely to come out of the K-fold sampling of $q(G)$.

As usual, one can use gradient descent and Monte Carlo sampling to minimize this Lagrangian, taking care to account for q's now appearing twice in the integrand of the $E(G)$ term. Note that the Monte Carlo process includes sampling the probability distribution $\frac{\Theta[\kappa - \int dx'\ q(x')\Theta[G(x)-G(x')]]}{\kappa}$ as well as the q_i. This means that only those points in the best κ'th percentile are kept, and used for all Monte Carlo estimates. This may cause greater noise in the Monte Carlo sampling than would be the case for $\kappa = 1$.

As an example, say that for agent i, all of its moves have the same value of $E(G \mid x_i)$, and similarly for agent j, and say that G is optimal if agents i and j both make move 0. Then if we modify the updating so that agent i only considers the best values that arose when it made move 0, and similarly for agent j, then both will be steered to prefer to make move 0 to their alternatives. This will cause them to coordinate their moves in an optimal manner.

A similar modification is to replace G with $f(G)$ in the Maxent Lagrangian, for some concave nowhere-decreasing function $f(.)$. This would distort G to accentuate those x's with good values. Intuitively, this will have the effect of coordinating the updates of the separate q_i at the end of the block, in a way to help lower G. The price paid for this is that there will be more variance in the values of $f(G)$ returned by the Monte Carlo sampling than those of G, in general.

Note that if q is a local minimum of the Lagrangian for G, in general it will not be a local minimum for the Lagrangian of $f(G)$ (the gradient will no longer be zero under that replacement, in general). So we can replace G with $f(G)$ when we get stuck in a local minimum, and then return to G once q gets away from that local minimum. In this way we can break out of local minima, without facing the penalty of extra variance. Of course, none of these advantages in replacing G with $f(G)$ hold for algorithms that directly search for an x giving a good $G(x)$ value; x is a local minimum of $G(x) \Leftrightarrow x$ is a local minimum of $f(G(x))$.

An even simpler modification to the $E(G)$ term than those considered above is to replace $G(x)$ with $\Theta[G(x)-K]$. Under this replacement the $E(G)$ term becomes the probability that $G(x) > K$. So minimizing it will push q to x with lower G values. For this modified Lagrangian, the gradient descent update step adds the following to each $q_i(x_i)$:

$$\alpha\Big[\beta q(G < K \mid x_i) + \ln(q_i(x_i)) - \frac{\sum_{x_i'} \beta q(G < K \mid x_i') + \ln(q_i(x_i'))}{\sum_{x_i'} 1}\Big].$$

In gradient descent of the Maxent Lagrangian we must Monte Carlo estimate the expected value of a real number (G). In contrast, in gradient descent of this modified Lagrangian we Monte Carlo estimate the expected value of a single bit: whether G exceeds K. Accordingly, the noise in the Monte Carlo estimation for this modified Lagrangian is usually far smaller.

In all these variants it may make sense to replace the Heaviside function with a logistic function or an exponential. In addition, in all of them the annealing schedule for K can be set by periodically searching for the K that is (estimated to be) optimal, just as one searches for optimal coordinate systems [1], [13]. Alternatively, a simple heuristic is to have K at the end of each block be set so that some pre-fixed percentage of the sampled points in the block go into our calculation of how to update q.

Yet another possibility is to replace $E(G)$ with the κ'th percentile G value, i.e., with the K such that $\int dx'\ q(x')\Theta(G(x') - K) = \kappa$. (To evaluate the partial derivative of that K with respect a particular $q_i(x_i)$ one must use implicit differentiation.)

B. KL-based Lagrangians

Both the qp-KL Lagrangian and pq-KL Lagrangians discussed above had the target distribution be a Boltzmann distribution over G. For high enough β, such a distribution is peaked near $\text{argmin}_x G(x)$. So sampling an accurate approximation to it should give an x with low G, if β is large enough. This is why one way to minimize G is to iteratively find a q that approximates the Boltzmann distribution, for higher and higher β.

However there are other target distributions that are peaked about minimizers of G. In particular, given any distribution p, the masked distribution

$$\mathcal{M}_{\Theta(K-G(.))}(p)(x) \triangleq \frac{p(x)\Theta[K - G(x)]}{\int dx'\ p(x')\Theta[K - G(x')]}$$

is guaranteed to be more peaked about such minimizers than is p. So our minimization can be done by iterating the process of finding the q that best approximates $\mathcal{M}_{\Theta(K-G(.))}(p)$ and then setting $p = q$. This is analogous to the minimization algorithm considered in previous sections, which

iterates the process of finding the q that best approximates the Boltzmann distribution and then increases β.

For the choice of pq-KL distance as the approximation error, the q that best approximates $\mathcal{M}_{\Theta(K-G(.))}(p)$ is just the product of the marginal distributions of $\mathcal{M}_{\Theta(K-G(.))}(p)$. So at the end of each iteration, we replace

$$\begin{aligned} q_i(x_i) &\leftarrow \frac{\int dx'_{(i)} q'(x'_{(i)}, x_i)\Theta[K - G(x_i, x'_{(i)})]}{\int dx'\ q'(x')\Theta[K - G(x')]} \\ &= \frac{q'(G < K, x_i)}{q'(G < K)} \\ &= q'(x_i \mid G < K) \end{aligned}$$

where q' is the product distribution being replaced. This can be Monte-Carlo estimated by agent i using only observed G values in the usual way. So like gradient descent on the Maxent Lagrangian, this update rule is well-suited to a distributed implementation.

Note that if we replace the Heaviside function masking function with a "softened version" like a logistic function with exponent β about K, $\tilde{\Theta}_{\beta,K}(x) \triangleq [1 + e^{\beta(G(x)-K)}]^{-1}$, the update rule becomes

$$q_i(x_i) \leftarrow \frac{E(\tilde{\Theta}_{\beta,K} \mid x_i)q'(x_i)}{E(\tilde{\Theta}_{\beta,K})}. \tag{7}$$

Similarly, we can replace the Heaviside function with a Bolzmann distribution with exponent β, getting the update rule

$$q_i(x_i) \leftarrow \frac{E(e^{-\beta G} \mid x_i)q'(x_i)}{E(e^{-\beta G})}. \tag{8}$$

where both expectations are evaluated under q', the distribution that generated the Monte Carlo samples. It's interesting to compare this update rule with the parallel Brouwer update rule for the team game [21], [13], [14], to which it is very similar. In parallel Brouwer updating, each i adopts the new distribution q_i^* that would minimize its Maxent Lagrangian *if all other agents did not change their distributions*, as given by Eq. 2.[14] This is done simultaneously by all agents. Now in general, when any q_i updates, what distribution q_j is optimal for any $j \neq i$ will change.[15] Accordingly, a step of parallel Brouwer updating can "thrash", and not actually decrease the associated (Maxent) Lagrangian. In contrast, the update in Eq. 8 is guaranteed to minimize its associated Lagrangian of pq distance to $\mathcal{M}_{\Theta(K-G(.))}(q')$. On the other hand, since it is based on the pq-KL Lagrangian to $\mathcal{M}_{\Theta(K-G(.))}(q')$ rather than on $\mathcal{M}_{\Theta(K-G(.))}(q')$ directly, it is not clear how minimization of this Lagrangian affects the ultimate object of interest, $E(G)$. Note also that this update rule is also very similar to the adaptive importance sampling of the original pq-KL approach discussed in [13].

[14]In practice that update is estimated via Monte-Carlo samples, in the usual way. Accordingly, parallel Brouwer updating update can be viewed as a variant of empirical best response game-playing.

[15]This can be partly mitigated by not having each i adopt the exact distribution q_i^*, but rather jump part way from the current q_i to q_i^*.

Finally, rather than pq KL distance, consider using qp KL distance to approximate the Boltzmann-masked distribution, $q'(x)\frac{e^{-\beta G(x)}}{\int dx'\ q'(x')e^{-\beta G(x')}}$. Up to an overall additive constant, that distance is just the Maxent Lagrangian, with one difference: $S(q) \rightarrow -KL(q \ \| \ q')$. This means the temperature is effectively 0 in subsequent search. For example, the first order descent direction for this new Lagrangian, evaluated at the current point, q', has components $\beta E_{q'}(G \mid x_i)$ for all i and x_i, with no $\ln q_i(x_i)$ term.[16]

On the other hand, given the current q', the optimal solution Eq. 2 changes to

$$q_i^\beta(x_i) \propto q'_i(x_i)e^{-\beta E_{q^\beta_{(i)}}(G|x_i)} \tag{9}$$

where as before q' is the current distribution. So Brouwer updating is now different, with the current distribution q' serving as a prior probability μ (see Eq. 1). Evaluation of gradients and Brouwer updates for qp KL distance for a logistic masking function proceeds similarly. (We get the same formulas, just with $G(x)$ replaced throughout by $\frac{\ln[1+e^{\beta(G(x)-K)}]}{\beta}$.)

It is interesting to compare the multiplicative updating of Eq. 9 with that of Nearest Newton. If we expand the exponential in Eq. 9 to first order and ignore the *ln* term of Nearest Newton (i.e., take temperature to 0, with stepsize changed accordingly), the two kinds of update become almost identical. (The remaining difference is that Nearest Newton normalizes the update to stay in $\mathcal{P}$ by adding a normalizing vector rather than by dividing by a normalizing scalar.) This connection is not too surprising, in light of the fact that in the continuum time limit with data-aging, Nearest Newton and parallel Brouwer updating become identical, with the stepsize of the Nearest Newton identically equal to the aging-constant in the parallel Brouwer [12].

V. Conclusion

Many problems in adaptive, distributed control can be cast as an iterated game. The coupling between the mixed strategies of the players arises as the system evolves from one instant to the next. This is what the system designer determines. Information theory tells us that the most likely joint strategy of the players, given a value of the expectation of the overall control objective function, is the minimizer of a particular Lagrangian function of the joint strategy. So the goal of the system designer is to speed evolution of the joint strategy to that Lagrangian minimizing point, lower the expectated value of the control objective function, and repeat. Here we elaborate the theory of algorithms that do this using local descent procedures, and that thereby achieve efficient, adaptive, distributed control.

VI. Appendix

This appendix provides proofs absent from the main text.

[16]It is interesting to consider having the distribution we wish to mask not be the current distribution, but rather some previous one. In this case the descent is based on both the current distribution and the previous one, i.e., the descent rule gives a second order dynamic system.

A. Derivation of Lemma 1

Proof: Consider the set of $\vec{u}$ such that the directional derivatives $D_{\vec{u}} f_i$ evaluated at y' all equal 0. These are the directions consistent with our constraints to first order. We need to find the one of those $\vec{u}$ such that $D_{\vec{u}} V$ evaluated at y' is maximal.

To simplify the analysis we introduce the constraint that $|\vec{u}| = 1$. This means that the directional derivative $D_{\vec{u}} V$ for any function V is just $\vec{u} \cdot \nabla V$. We then use Lagrange parameters to solve our problem. Our constraints on $\vec{u}$ are $\sum_j u_j^2 = 1$ and $D_{\vec{u}} f_i(y') = \vec{u} \cdot \nabla f_i(y') = 0 \ \ \forall i$. Our objective function is $D_{\vec{u}} V(y') = \vec{u} \cdot \nabla V(y')$.

Differentiating the Lagrangian gives

$$2\lambda_0 u_i + \sum_i \lambda_i \nabla f = \nabla V \ \ \forall i,$$

where all dependencies on y' are implicit. The solution is

$$u_i = \frac{\nabla V - \sum_i \lambda_i \nabla f}{2\lambda_0}.$$

λ_0 enforces our constraint on $|\vec{u}|$. Since we are only interested in specifying $\vec{u}$ up to a proportionality constant, we can set $2\lambda_0 = 1$. Redefining the Lagrange parameters by multiplying them by -1 then gives the result claimed. **QED.**

B. Proof of claims following Lemma 1

For generality, the proofs provided here allow the private utilities g_i to differ from one another. See the discussion in Section III-C.

i) Define $f_i(q) \triangleq \int dx_i q_i(x_i)$, i.e., f_i is the constraint forcing q_i to be normalized. Now for any q that equals zero for some joint move there must be an i and an x_i' such that $q_i(x_i') = 0$. Plugging into Lemma 1, we can evaluate the component of the direction of steepest descent along the direction of player i's probability of making move x_i':

$$\frac{\partial L}{\partial q_i(x_i)} + \lambda \frac{\partial f_i}{\partial q_i(x_i)} = \beta E(g_i \mid x_i) + \ln(q_i(x_i)) - \frac{\int dx_i'' [\beta E(g_i \mid x_i'') + \ln(q_i(x_i''))]}{\int dx_i'' 1}$$

Since there must some x_i'' such that $q_i(x_i'') \neq 0$, $\exists x_i$ such that $\beta E(g_i \mid x_i'') + \ln(q_i(x_i''))$ is finite. Therefore our component is negative infinite. So L can be reduced by increasing $q_i(x_i')$. Accordingly, no q having zero probability for some joint move x can be a minimum of i's Lagrangian.

ii) To construct a bounded rational game with multiple equilibria, note that at any (necessarily interior) local minimum q, for each i,

$$\beta E(g_i \mid x_i) + \ln(q_i(x_i)) = \beta \int dx_{(i)} g_i(x_i, x_{(i)}) \prod_{j \neq i} q_j(x_j) + \ln(q_i(x_i))$$

must be independent of x_i, by Lemma 1. So say there is a component-by-component bijection $T(x) \triangleq (T_1(x_1), T_2(x_2), \ldots)$ that leaves all the $\{g_j\}$ unchanged, i.e., such that $g_j(x) = g_j(T(x)) \ \forall x, j$ [17].

Define q' by $q'(x) = q(T(x)) \ \forall x$. Then for any two values x_i^1 and x_i^2,

$$\begin{aligned}
&\beta E_{q'}(g_i \mid x_i^1) + \ln(q_i'(x_i^1)) \\
&\quad - \beta E_{q'}(g_i \mid x_i^2) \ + \ \ln(q_i'(x_i^2)) \\
&\quad = \\
&\beta \int dx_{(i)} g_i(x_i^1, x_{(i)}) \prod_{j \neq i} q_j(T(x_j)) \ + \ \ln(q_i(T(x_i^1))) \\
&\quad - \beta \int dx_{(i)} g_i(x_i^2, x_{(i)}) \prod_{j \neq i} q_j(T(x_j))) \\
&\qquad + \ln(q_i(T(x_i^2))) \\
&\quad = \\
&\beta \int dx_{(i)} g_i(x_i^1, T^{-1}(x_{(i)})) \prod_{j \neq i} q_j(x_j) \ + \ \ln(q_i(T(x_i^1))) \\
&\quad - \beta \int dx_{(i)} g_i(x_i^2, T^{-1}(x_{(i)})) \prod_{j \neq i} q_j(x_j)) \\
&\qquad + \ln(q_i(T(x_i^2))) \\
&\quad = \\
&\beta \int dx_{(i)} g_i(T(x_i^1), x_{(i)})) \prod_{j \neq i} q_j(x_j) \ + \ \ln(q_i(T(x_i^1))) \\
&\quad - \beta \int dx_{(i)} g_i(T(x_i^2), x_{(i)})) \prod_{j \neq i} q_j(x_j)) \\
&\qquad + \ln(q_i(T(x_i^2))) \\
&\quad = \\
&\beta E_q(g_i \mid T(x_i^1)) \ + \ \ln(q_i(T(x_i^1))) \\
&\quad - \beta E_q(g_i \mid T(x_i^2)) \ + \ \ln(q_i(T(x_i^2)))
\end{aligned}$$

where the invariance of g_i was used in the penultimate step. Since q is a local minimum though, this last difference must equal 0. Therefore q' is also a local minimum.

Now choose the game so that $\forall i, x_i, T(x_i) \neq x_i$. (Our congestion game example has this property.) Then the only way the transformation $q \to q(T)$ can avoiding producing a new product distribution is if $q_i(x_i) = q_i(x_i') \ \forall i, x_i, x_i'$, i.e., q is uniform. Say the Hessians of the players' Lagrangians are not all positive definite at the uniform q. (For example have our congestion game be biased away from uniform multiplicities.) Then that q is not a local minimum of the Lagrangians. Therefore at a local minimum, $q \neq q(T)$. Accordingly, q and $q(T)$ are two distinct equilibria.

iii) To establish that at any q there is always a direction along which any player's Lagrangian is locally convex, fix

[17] As an example, consider a congestion team game. In such a game all players have the same set of possible moves, and the shared utility G is a function only of the k-indexed bit string $\{N(x,k)\}$, where $N(x,k) = 1$ iff there is a move that is shared by exactly k of the players when the joint move is x. In this case T just permutes the set of possible moves in the same way for all players.

all but two of the $\{q_i\}$, q_0 and q_1, and fix both q_0 and q_1 for all but two of their respective possible values, which we can write as $q_0(0), q_0(1), q_1(0)$, and $q_1(1)$, respectively. So we can parameterize the set of q we're considering by two real numbers, $x \triangleq q_0(0)$ and $y \triangleq q_1(0)$. The 2×2 Hessian of L as a function of x and y is

$$\begin{pmatrix} \frac{1}{x} + \frac{1}{a-x} & \alpha \\ \alpha & \frac{1}{y} + \frac{1}{b-y} \end{pmatrix}$$

where $a \triangleq 1 - q_0(0) - q_0(1)$ and $b \triangleq 1 - q_1(0) - q_1(1)$, and α is a function of g_i and $\prod_{j \neq 0,1} q_j$. Defining $s \triangleq \frac{1}{x} + \frac{1}{a-x}$ and $t \triangleq \frac{1}{y} + \frac{1}{b-y}$, the eigenvalues of that Hessian are

$$\frac{s + t \pm \sqrt{4\alpha^2 + (s-t)^2}}{2}.$$

The eigenvalue for the positive root is necessarily positive. Therefore along the corresponding eigenvector, L is convex at q. **QED.**

C. Dependence of $E_q(G)$ on β

i) There are several ways to show that the value of $E_{q_i^\beta}([g_i]_{i,q_{(i)}})$ must shrink as β grows. Here we do so by evaluating the associated derivative with respect to β.

Define $N(U) \triangleq \int dy\, e^{-U(y)}$, the normalization constant for the distribution proportional to $e^{-U(y)}$. View the x_i-indexed vector q_i^β as a function of β, g_i and $q_{(i)}$. So we can somewhat inelegantly write $E(g_i) = E_{q_i^\beta(\beta, g_i, q_{(i)}), q_{(i)}}(g_i)$. Then one can expand

$$\begin{aligned} \frac{\partial E(g_i)}{\partial \beta} &= -\frac{\partial^2 \ln(N(\beta [g_i]_{i,q_{(i)}}))}{\partial \beta^2} \\ &= -\mathrm{Var}([g_i]_{i,q_{(i)}}) \end{aligned}$$

where the variance is over possible x_i, sampled according to $q_i^\beta(x_i)$. **QED.**

ii) In general, there are multiple solutions to Eq. 2, at all of which $\vec{\nabla} L(q) = \vec{0}$. So to analyze the β-dependence of the expected value of G one has to keep track of each of those multiple solutions and their separate dependences on β. This can be quite laborious.

As an alternative, recall from the text that there is a single fully coupled distribution p^β that minimizes $L(p) \triangleq \beta(E_p(G) - \gamma) - S$; this is the optimal p. In addition, as discussed in the text, the Maxent Lagrangian $L(q)$ is equal to the Kullback-Leibler distance from q to that optimal p. So the solutions of Eq. 2 are the q's that locally minimize the distance to p^β. So those q's "track" the distribution p^β whenever it moves due to a change in β.

Now as shown elsewhere in this appendix, $E_{p^\beta}(G)$ is a decreasing function of β. So when we increase β, the associated solutions of Eq. 2 track p^β as closely as possible as it lowers its $E(G)$. In this sense, raising β is "equivalent" to lowering expected G for the solutions to Eq. 2.

References

[1] D. H. Wolpert, "Information theory — the bridge connecting bounded rational game theory and statistical physics," in *Complex Engineering Systems*, A. M. D. Braha and Y. Bar-Yam, Eds., 2004.

[2] D. Laughlin, M. Morari, and R. Braatz, "Robust performance of cross-directional control systems for web processes," *Automatica*, vol. 29, pp. 1394–1410, 1993.

[3] J. Wolfe, D. Chichka, and J. Speyer, "Decentralized controllers for unmanned aerial vehicle formation flight," *American Institute of Aeronautics and Astronautics*, vol. 96, p. 3933, 1996.

[4] M. Mesbai and F. Hadaegh, "Graphs, matrix inequalities, and switching for the formation flying control of multiple spacecraft," in *Proceedings of the American Control Conference, San Diego, CA*, 1999, pp. 4148–4152.

[5] D. Fudenberg and J. Tirole, *Game Theory*. Cambridge, MA: MIT Press, 1991.

[6] T. Basar and G. Olsder, *Dynamic Noncooperative Game Theory*. Philadelphia, PA: Siam, 1999, second Edition.

[7] M. Osborne and A. Rubenstein, *A Course in Game Theory*. Cambridge, MA: MIT Press, 1994.

[8] R. Aumann and S. Hart, *Handbook of Game Theory with Economic Applications*. North-Holland Press, 1992.

[9] D. Fudenberg and D. K. Levine, *The Theory of Learning in Games*. Cambridge, MA: MIT Press, 1998.

[10] D. H. Wolpert and S. Bieniawski, "Adaptive distributed control: beyond single-instant categorical variables," in *Proceedings of MSRAS04*, A. S. et al, Ed. Springer Verlag, 2004.

[11] D. H. Wolpert, "Bounded rationality game theory and information theory," 2004, submitted.

[12] ——, "What information theory says about best response, binding contracts, and collective intelligence," in *Proceedings of WEHIA04*, A. N. et al, Ed. Springer Verlag, 2004.

[13] ——, "Factoring a canonical ensemble," 2003, cond-mat/0307630.

[14] S. Bieniawski and D. H. Wolpert, "Using product distributions for distributed optimization," in *Proceedings of ICCS 04*, 2004.

[15] D. H. Wolpert and C. F. Lee, "Adaptive metropolis hastings sampling using product distributions," in *Proceedings of ICCS 04*, 2004.

[16] D. H. Wolpert, "Finding bounded rational equilibria part i: General techniques," in *Proceedings of the International Society of Dynamic Games Conference, 2004*, 2004, in press.

[17] W. Macready, S. Bieniawski, and D. Wolpert, "Adaptive multi-agent systems for constrained optimization," 2004, technical report IC-04-123.

[18] N. Antoine, S. Bieniawski, I. Kroo, and D. H. Wolpert, "Fleet assignment using collective intelligence," in *Proceedings of 42nd Aerospace Sciences Meeting*, 2004, aIAA-2004-0622.

[19] C. F. Lee and D. H. Wolpert, "Product distribution theory for control of multi-agent systems," in *Proceedings of AAMAS 04*, 2004.

[20] S. Airiau and D. H. Wolpert, "Product distribution theory and semi-coordinate transformations," 2004, submitted to AAMAS 04.

[21] S. Bieniawski and D. H. Wolpert, "Adaptive, distributed control of constrained multi-agent systems," in *Proceedings of AAMAS04*, 2004.

[22] S. Bieniawski, D. H. Wolpert, and I. Kroo, "Discrete, continuous, and constrained optimization using collectives," in *Proceedings of 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York*, 2004, in press.

[23] T. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley-Interscience, 1991.

[24] D. Mackay, *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[25] E. T. Jaynes and G. L. Bretthorst, *Probability Theory : The Logic of Science*. Cambridge University Press, 2003.

[26] S. Kirkpatrick, C. D. J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May 1983.

[27] R. Diekmann, R. Luling, and J. Simon, "Problem independent distributed simulated annealing and its applications," in *Applied Simulated Annealing*. Springer, 1993, pp. 17–44.

[28] O. Catoni, "Solving scheduling problems by simulated annealing," *SIAM Journal on Control and Optimization*, vol. 36, no. 5, pp. 1539–1575, 1998.

[29] R. V. V. Vidal, Ed., *Applied Simulated Annealing (Lecture Notes in Economics and Mathematical Systems)*. Springer, 1993.

[30] D. H. Wolpert and S. Bieniawski, "Distributed control by lagrangian steepest descent," in *Proceedings of CDC04*, 2004.
[31] D. H. Wolpert and W. Macready, "Self-dissimilarity as a high dimensional complexity measure," in *Proceedings of the International Conference on Complex Systems, 2004*, 2004, in press.
[32] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd ed.)*. Wiley and Sons, 2000.